

Semantic Web Modelling Languages (Part 2)

Tutorial at IJCAI-09
July 13, 2009



Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft



Universität Karlsruhe (TH)
Research University • founded 1825



Pascal Hitzler



Markus Krötzsch



Sebastian Rudolph

AIFB, Universität Karlsruhe (TH)
Germany

<http://www.pascal-hitzler.de>

<http://korrekt.org>

<http://www.sebastian-rudolph.de>

Full set of slides available from

http://semantic-web-grundlagen.de/wiki/IJCAI-09_Tutorial

■ Web Ontology Language

- W3C Recommendation for the Semantic Web, 2004
- OWL 2 (revised W3C Recommendation) forthcoming in 2009
– we already present this here

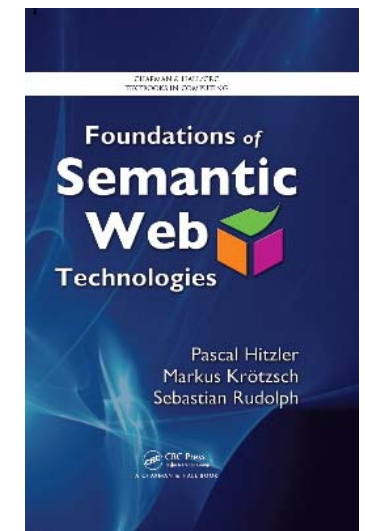
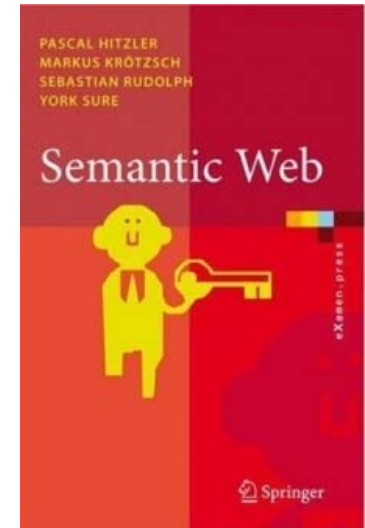
■ Semantic Web KR language based on description logics (DLs)

- OWL DL is essentially DL SROIQ(D)
- KR for web resources, using URIs.
- Using web-enabled syntaxes, e.g. based on XML or RDF.
We present
 - DL syntax (used in research – not part of the W3C recommendation)
 - RDF Turtle syntax

- **W3C OWL Working Group, OWL 2 Web Ontology Language: Document Overview. <http://www.w3.org/TR/owl2-overview/>**
- **Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter Patel-Schneider, Sebastian Rudolph, OWL 2 Web Ontology Language: Primer. <http://www.w3.org/TR/owl2-primer/>**
- **Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider, The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2nd edition, 2007.**

References – Textbooks

- **Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure, Semantic Web – Grundlagen.**
Springer, 2008.
<http://www.semantic-web-grundlagen.de/>
(In German.)
- **Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, Foundations of Semantic Web Technologies.**
Chapman & Hall/CRC, 2009.
<http://www.semantic-web-book.org/wiki/FOST>
(Ask for a flyer from us.)



- **OWL – Basic Ideas**
- **OWL As the Description Logic SROIQ(D)**
- **Different Perspectives on OWL**
- **Expressivity Examples: Rules in OWL**
- **OWL Semantics**
- **OWL Profiles**
- **Proof Theory**
- **Tools**
- **Outlook**
- **References**

- **OWL – Basic Ideas**
- **OWL As the Description Logic SROIQ(D)**
- **Different Perspectives on OWL**
- **Expressivity Examples: Rules in OWL**
- **OWL Semantics**
- **OWL Profiles**
- **Proof Theory**
- **Tools**
- **Outlook**
- **References**

Rationale behind OWL

- Open World Assumption
- Favourable trade-off between expressivity and scalability
- Integrates with RDFS
- Purely declarative semantics

Features:

- Fragment of first-order predicate logic (FOL)
- Decidable
- Known complexity classes (N2ExpTime for OWL 2 DL)
- Reasonably efficient for real KBs

- **individuals (written as URIs)**
 - also: constants (FOL), resources (RDF)
 - `http://example.org/sebastianRudolph`
 - `http://www.semantic-web-book.org`
 - we write these lowercase and abbreviated, e.g. "sebastianRudolph"
- **classes (also written as URIs!)**
 - also: concepts, unary predicates (FOL)
 - we write these uppercase, e.g. "Father"
- **properties (also written as URIs!)**
 - also: roles (DL), binary predicates (FOL)
 - we write these lowercase, e.g. "hasDaughter"

DL syntax

FOL syntax

ABox statements

■ **Person(mary)**

■ **Person(mary)**

■ **Woman \sqsubseteq Person**

■ $\forall x (\text{Woman}(x) \rightarrow \text{Person}(x))$

■ **Person \equiv HumanBeing**

■ **hasWife(john,mary)**

■ **hasWife(john,mary)**

■ **hasWife \sqsubseteq hasSpouse**

■ $\forall x \forall y (\text{hasWife}(x,y) \rightarrow \text{hasSpouse}(x,y))$

■ **hasSpouse \equiv marriedWith**

TBox statements

DL syntax

RDFS syntax

■ Person(mary)

■ :mary rdf:type :Person .

■ Woman v Person

■ :Woman rdfs:subClassOf :Person .

■ Person \equiv HumanBeing

■ hasWife(john,mary)

■ :john :hasWife :mary .

■ hasWife v hasSpouse

■ :hasWife rdfs:subPropertyOf :hasSpouse .

■ hasSpouse \equiv marriedWith

Special classes and properties

- **owl:Thing** (RDF syntax)
 - DL-syntax: \top
 - contains everything
- **owl:Nothing** (RDF syntax)
 - DL-syntax: \perp
 - empty class
- **owl:topProperty** (RDF syntax)
 - DL-syntax: \mathcal{U}
 - every pair is in \mathcal{U}
- **owl:bottomProperty** (RDF syntax)
 - empty property

■ conjunction

$$\forall x (\text{Mother}(x) \leftrightarrow \text{Woman}(x) \wedge \text{Parent}(x))$$

- $\text{Mother} \equiv \text{Woman} \sqcap \text{Parent}$
- `:Mother owl:equivalentClass _:x .`
`_:x rdf:type owl:Class .`
`_:x owl:intersectionOf (:Woman :Parent) .`

■ disjunction

$$\forall x (\text{Parent}(x) \leftrightarrow \text{Mother}(x) \vee \text{Father}(x))$$

- $\text{Parent} \equiv \text{Mother} \sqcup \text{Father}$
- `:Parent owl:equivalentClass _:x .`
`_:x rdf:type owl:Class .`
`_:x owl:unionOf (:Mother :Father) .`

■ negation

$$\forall x (\text{ChildlessPerson}(x) \leftrightarrow \text{Person}(x) \wedge \neg \text{Parent}(x))$$

- $\text{ChildlessPerson} \equiv \text{Person} \sqcap \neg \text{Parent}$
- `:ChildlessPerson owl:equivalentClass _:x .`
`_:x rdf:type owl:Class .`
`_:x owl:intersectionOf (:Person _:y) .`
`_:y owl:complementOf :Parent .`

■ existential quantification

- only to be used with a role – also called a *property restriction*

- $\text{Parent} \equiv \exists \text{hasChild}.\text{Person}$

- `:Parent owl:equivalentClass _:x .`
`_:x rdf:type owl:Restriction .`
`_:x owl:onProperty :hasChild .`
`_:x owl:someValuesFrom :Person .`

$$\forall x (\text{Parent}(x) \leftrightarrow \exists y (\text{hasChild}(x,y) \wedge \text{Person}(y)))$$

■ universal quantification

- only to be used with a role – also called a *property restriction*

- $\text{Person} \sqcap \text{Happy} \equiv \forall \text{hasChild}.\text{Happy}$

- `_:x rdf:type owl:Class .`
`_:x owl:intersectionOf (:Person :Happy) .`
`_:x owl:equivalentClass _:y .`
`_:y rdf:type owl:Restriction .`
`_:y owl:onProperty :hasChild .`
`_:y owl:allValuesFrom :Happy .`

$$\forall x (\text{Person}(x) \wedge \text{Happy}(x) \leftrightarrow \forall y (\text{hasChild}(x,y) \rightarrow \text{Happy}(y)))$$

■ Class constructors can be nested arbitrarily

- OWL – Basic Ideas
- **OWL As the Description Logic SROIQ(D)**
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- OWL Semantics
- OWL Profiles
- Proof Theory
- Tools
- Outlook
- References

Understanding SROIQ(D)

The description logic ALC

Complexity: ExpTime

- ABox expressions:
Individual assignments
Property assignments

Father(john)
hasWife(john,mary)

- TBox expressions
subclass relationships \sqsubseteq
- conjunction \sqcap
- disjunction \sqcup
- negation \neg
- property restrictions \forall
 \exists

Also: \top , \perp

Understanding SROIQ(D)

ALC + role chains = SR

■ **hasParent o hasBrother \sqsubseteq hasUncle**

■ **:hasUncle owl:propertyChainAxiom (:hasParent :hasBrother) .**

$$\forall x \forall y (\exists z ((\text{hasParent}(x,z) \wedge \text{hasBrother}(z,y)) \rightarrow \text{hasUncle}(x,y)))$$

■ includes top property and bottom property

■ includes **S = ALC + transitivity**

■ **hasAncestor o hasAncestor \sqsubseteq hasAncestor**

■ includes **SH = S + role hierarchies**

■ **hasFather \sqsubseteq hasParent**

I'll skip RDF syntax in the following.

Understanding SROIQ(D)

- **O – nominals (closed classes)**
 - $\text{MyBirthdayGuests} \equiv \{\text{bill}, \text{john}, \text{mary}\}$
 - Note the difference to
 $\text{MyBirthdayGuests}(\text{bill})$
 $\text{MyBirthdayGuests}(\text{john})$
 $\text{MyBirthdayGuests}(\text{mary})$
- **Individual equality and inequality (no unique name assumption!)**
 - $\text{bill} = \text{john}$
 - $\{\text{bill}\} \equiv \{\text{john}\}$
 - $\text{bill} \neq \text{john}$
 - $\{\text{bill}\} \sqcap \{\text{john}\} \equiv \perp$
- **Complexity SHIQ: ExpTime. Complexity SHOIQ: NExpTime**
Complexity SROIQ: N2ExpTime

Understanding SROIQ(D)

■ I – inverse roles

- $\text{hasParent} \equiv \text{hasChild}^{-}$
- $\text{Orphan} \equiv \forall \text{hasChild}^{-}.\text{Dead}$

■ Q – qualified cardinality restrictions

- $\leq 4 \text{ hasChild.Parent}(\text{john})$
- $\text{HappyFather} \equiv \geq 2 \text{ hasChild.Female}$
- $\text{Car} \sqsubseteq =4 \text{ hasTyre.T}$

Understanding SROIQ(D)

Properties can be declared to be

■ Transitive	hasAncestor
■ Symmetric	hasSpouse
■ Asymmetric	hasChild
■ Reflexive	hasRelative
■ Irreflexive	parentOf
■ Functional	hasHusband
■ InverseFunctional	hasHusband

called *property characteristics*

(D) – datatypes

- so far, we have only seen properties with individuals in second argument, called *object properties* or *abstract roles* (DL)
- properties with datatype literals in second argument are called *data properties* or *concrete roles* (DL)
- allowed are many XML Schema datatypes, including `xsd:integer`, `xsd:string`, `xsd:float`, `xsd:boolean`, `xsd:anyURI`, `xsd:dateTime`

and also e.g. `owl:real`

(D) – datatypes

- `hasAge(john, "51"^^xsd:integer)`
- additional use of *constraining facets* (from XML Schema)
 - e.g. `Teenager \equiv Person $\sqcap \exists \text{hasAge} . (\text{xsd:integer} : \geq 12 \text{ and } \leq 19)$`
note: this is not standard DL notation!
 - `:Teenager rdfs:subClassOf _:x .`
`_:x rdf:type owl:Restriction .`
`_:x owl:onProperty :hasAge .`
`_:x owl:someValuesFrom _:y .`
`_:y rdf:type rdfs:Datatype .`
`_:y owl:onDatatype xsd:integer .`
`_:y owl:withRestrictions (`
`[xsd:minInclusive "13"^^xsd:integer]`
`[xsd:maxInclusive "19"^^xsd:integer]) .`

further expressive features

- **Self**
 - $\text{PersonCommittingSuicide} \equiv \exists \text{kills.Self}$
- **Keys (not really in SROIQ(D), but in OWL)**
 - set of (object or data) properties whose values uniquely identify an object
- **disjoint properties**
 - $\text{Disjoint}(\text{hasParent}, \text{hasChild})$
- **explicit anonymous individuals**
 - as in RDF: can be used instead of named individuals

SROIQ(D) constructors – overview

- ABox assignments of individuals to classes or properties
- ALC: \sqsubseteq, \equiv for classes
 $\sqcap, \sqcup, \neg, \exists, \forall$
 \top, \perp
- SR: + **property chains, property characteristics, role hierarchies** \sqsubseteq
- SRO: + nominals $\{n\}$
- SROI: + inverse properties
- SROIQ: + **qualified** cardinality constraints
- SROIQ(D): + datatypes (including **facets**)

- + **top and bottom roles** (for objects and datatypes)
- + **disjoint properties**
- + **Self**
- + **Keys** (not in SROIQ(D), but in OWL)

Some Syntactic Sugar in OWL

This applies to the non-DL syntaxes (e.g. RDF syntax).

- **disjoint classes**

- $\text{Apple} \sqcap \text{Pear} \sqsubseteq \perp$

- **disjoint union**

- $\text{Parent} \equiv \text{Mother} \sqcup \text{Father}$
 $\text{Mother} \sqcap \text{Father} \sqsubseteq \perp$

- **negative property assignments (also for datatypes)**

- $\neg \text{hasAge}(\text{jack}, "53"^^\text{xsd:integer})$

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- **Different Perspectives on OWL**
- Expressivity Examples: Rules in OWL
- OWL Semantics
- OWL Profiles
- Proof Theory
- Tools
- Outlook
- References

- OWL ontologies have URIs and can be referenced by others via
 - import statements
- Namespace declarations
- Entity declarations (must be done)
- Versioning information etc.

- Annotations
 - Entities and axioms (statements) can be endowed with annotations, e.g. using `rdfs:comment`.
 - OWL syntax provides *annotation properties* for this purpose.

The modal logic perspective

- Description logics can be understood from a modal logic perspective.
- Each pair of $\forall R$ and $\exists R$ statements give rise to a pair of modalities.
- Essentially, some description logics are multi-modal logics.
- See [The Description Logic Handbook].

The RDFS perspective

- `:mary rdf:type :Person .`
- `:Mother rdfs:subClassOf :Woman .`
- `:john :hasWife :Mary .`
- `:hasWife rdfs:subPropertyOf :hasSpouse`
- `:hasWife rdfs:range :Woman .`
- `:hasWife rdfs:domain :Man .`
- `Person(mary)`
- `Mother \sqsubseteq Woman`
- `hasWife(john,mary)`
- `hasWife \sqsubseteq hasSpouse`
- $\top \sqsubseteq \forall \text{hasWife. Woman}$
- $\top \sqsubseteq \forall \text{hasWife}^{\neg} . \text{Man}$ or $\exists \text{hasWife. } \top \sqsubseteq \text{Man}$

RDFS also allows to

- make statements about statements
→ only possible through annotations in OWL
- mix class names, individual names, property names (they are all URIs)
→ *punning* in OWL

- Description logics impose *type separation*, i.e. names of individuals, classes, and properties must be disjoint.
- In OWL 2 Full, type separation does not apply.
- In OWL 2 DL, type separation is relaxed, but a class X and an individual X are interpreted semantically as if they were different.
- See further below on the two different semantics for OWL.

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- **Expressivity Examples: Rules in OWL**
- OWL Semantics
- OWL Profiles
- Proof Theory
- Tools
- Outlook
- References

■ $\text{Man}(x) \wedge \text{hasBrother}(x,y) \wedge \text{hasChild}(y,z) \rightarrow \text{Uncle}(x)$

■ $\text{Man} \sqcap \exists \text{hasBrother}.\exists \text{hasChild}.\top \sqsubseteq \text{Uncle}$

■ $\text{kills}(x,x) \rightarrow \text{suicide}(x)$

■ $\exists \text{kills}.\text{Self} \sqsubseteq \text{suicide}$

$\text{suicide}(x) \rightarrow \text{kills}(x,x)$

$\text{suicide} \sqsubseteq \exists \text{kills}.\text{Self}$

Note: with these two axioms,

suicide is basically the same as *kills*

■ $\text{NutAllergic}(x) \wedge \text{NutProduct}(y) \rightarrow \text{dislikes}(x,y)$

■ $\text{NutAllergic} \equiv \exists \text{nutAllergic}.\text{Self}$

$\text{NutProduct} \equiv \exists \text{nutProduct}.\text{Self}$

$\text{nutAllergic} \circ \text{U} \circ \text{nutProduct} \sqsubseteq \text{dislikes}$

- $\text{dislikes}(x,z) \wedge \text{Dish}(y) \wedge \text{contains}(y,z) \rightarrow \text{dislikes}(x,y)$
 - $\text{Dish} \equiv \exists \text{dish}.\text{Self}$
 $\text{dislikes} \circ \text{contains}^- \circ \text{dish} \sqsubseteq \text{dislikes}$

- $\text{worksAt}(x,y) \wedge \text{University}(y) \wedge \text{supervises}(x,z) \wedge \text{PhDStudent}(z) \rightarrow \text{professorOf}(x,z)$
 - $\exists \text{worksAt}.\text{University} \equiv \exists \text{worksAt}.\text{University}.\text{Self}$
 $\text{PhDStudent} \equiv \exists \text{phDStudent}.\text{Self}$
 $\text{worksAt}.\text{University} \circ \text{supervises} \circ \text{phDStudent} \sqsubseteq \text{professorOf}$

- For more on OWL 2 and rules, see [Description Logic Rules] and [ELP].

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- **OWL Semantics**
- OWL Profiles
- Proof Theory
- Tools
- Outlook
- References

■ There are two semantics for OWL.

1. Description Logic Semantics

also: Direct Semantics; FOL Semantics

Can be obtained by translation to FOL.

Syntax restrictions apply! (see next slide)

2. RDF-based Semantics

No syntax restrictions apply.

Extends the direct semantics with RDFS-reasoning features.

In the following, we will deal with the direct semantics only.

To obtain decidability, syntactic restrictions apply.

- **Type separation / punning**
- **No cycles in property chains.**
- **No transitive properties in cardinality restrictions.**

- arbitrary property chain axioms lead to undecidability
- restriction: set of property chain axioms has to be *regular*
 - there must be a strict linear order $<$ on the properties
 - every property chain axiom has to have one of the following forms:

$R \pm R \sqsubseteq R$

$S^- \sqsubseteq R$

$S_1 \pm S_2 \pm \dots \pm S_n \sqsubseteq R$

$R \pm S_1 \pm S_2 \pm \dots \pm S_n \sqsubseteq R$

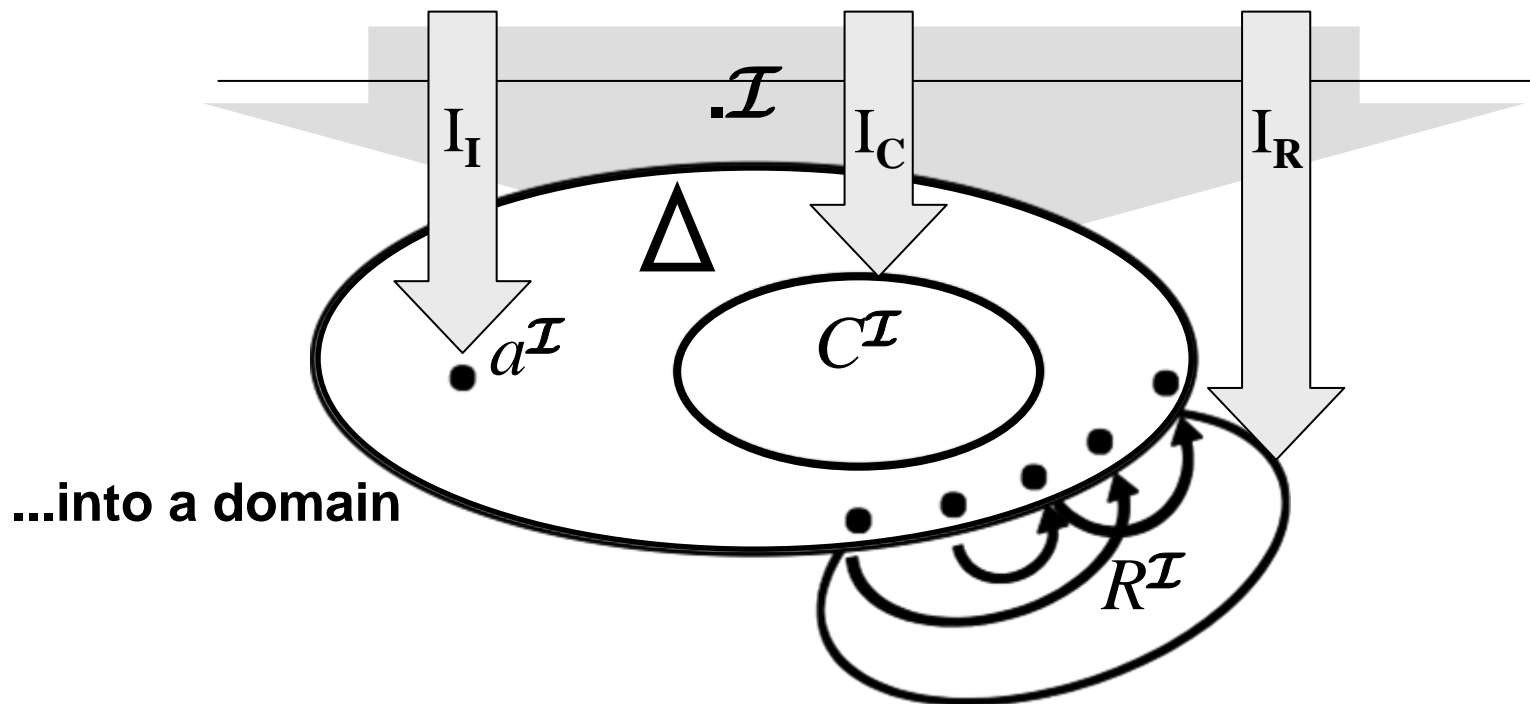
$S_1 \pm S_2 \pm \dots \pm S_n \pm R \sqsubseteq R$
 - thereby, $S_i < R$ for all $i = 1, 2, \dots, n$.
- Example 1: $R \pm S \sqsubseteq R \quad S \pm S \sqsubseteq S \quad R \pm S \pm R \sqsubseteq T$
→ regular with order $S < R < T$
- Example 2: $R \pm T \pm S \sqsubseteq T$
→ not regular because form not admissible
- Example 3: $R \pm S \sqsubseteq S \quad S \pm R \sqsubseteq R$
→ not regular because no adequate order exists

- combining property chain axioms and cardinality constraints may lead to undecidability
- restriction: use only *simple* properties in cardinality expressions (i.e. those which cannot be – directly or indirectly – inferred from property chains)
- technically:
 - for any property chain axiom $S_1 \pm S_2 \pm \dots \pm S_n \sqsubseteq R$ with $n > 1$, R is non-simple
 - for any subproperty axiom $S \sqsubseteq R$ with S non-simple, R is non-simple
 - all other properties are simple
- Example: $Q \pm P \sqsubseteq R$ $R \pm P \sqsubseteq R$ $R \sqsubseteq S$ $P \sqsubseteq R$ $Q \sqsubseteq S$
non-simple: R, S simple: P, Q

OWL Direct Semantics

- model-theoretic semantics
- starts with interpretations
- an interpretation maps

individual names, class names and property names...



■ mapping is extended to complex class expressions:

- $\top^I = \Delta^I$ $\perp^I = \emptyset$
- $(C \sqcap D)^I = C^I \cap D^I$ $(C \sqcup D)^I = C^I \cup D^I$ $(\neg C)^I = \Delta^I \setminus C^I$
- $\forall R.C = \{ x \mid \forall (x,y) \in R^I \rightarrow y \in C^I \}$
 $\exists R.C = \{ x \mid \exists (x,y) \in R^I \wedge y \in C^I \}$
- $\geq n R.C = \{ x \mid \#\{ y \mid (x,y) \in R^I \wedge y \in C^I \} \geq n \}$
- $\leq n R.C = \{ x \mid \#\{ y \mid (x,y) \in R^I \wedge y \in C^I \} \leq n \}$

■ ...and to role expressions:

- $U^I = \Delta^I \times \Delta^I$ $(R^-)^I = \{ (y,x) \mid (x,y) \in R^I \}$

■ ...and to axioms:

- $C(a)$ holds, if $a^I \in C^I$ $R(a,b)$ holds, if $(a^I, b^I) \in R^I$
- $C \sqsubseteq D$ holds, if $C^I \subseteq D^I$ $R \sqsubseteq S$ holds, if $R^I \subseteq S^I$
- $\text{Dis}(R,S)$ holds if $R^I \cap S^I = \emptyset$
- $S_1 \pm S_2 \pm \dots \pm S_n \sqsubseteq R$ holds if $S_1^I \pm S_2^I \pm \dots \pm S_n^I \subseteq R^I$

OWL Direct Semantics via FOL

- but often OWL 2 DL is said to be a fragment of FOL (with equality)...
- yes, there is a translation of OWL 2 DL into FOL

$$\begin{aligned}
 \pi(C \sqsubseteq D) &= (\forall x)(\pi_x(C) \rightarrow \pi_x(D)) \\
 \pi_x(A) &= A(x) \\
 \pi_x(\neg C) &= \neg \pi_x(C) \\
 \pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(D) \\
 \pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(D) \\
 \pi_x(\forall R.C) &= (\forall x_1)(R(x, x_1) \rightarrow \pi_{x_1}(C)) \\
 \pi_x(\exists R.C) &= (\exists x_1)(R(x, x_1) \wedge \pi_{x_1}(C)) \\
 \pi_x(\geq n S.C) &= (\exists x_1) \dots (\exists x_n) \left(\bigwedge_{i \neq j} (x_i \neq x_j) \wedge \bigwedge_i (S(x, x_i) \wedge \pi_{x_i}(C)) \right) \\
 \pi_x(\leq n S.C) &= \neg (\exists x_1) \dots (\exists x_{n+1}) \left(\bigwedge_{i \neq j} (x_i \neq x_j) \wedge \bigwedge_i (S(x, x_i) \wedge \pi_{x_i}(C)) \right) \\
 \pi_x(\{a\}) &= (x = a) \\
 \pi_x(\exists S.\text{Self}) &= S(x, x) \\
 \pi(R_1 \sqsubseteq R_2) &= (\forall x)(\forall y)(\pi_{x,y}(R_1) \rightarrow \pi_{x,y}(R_2)) \\
 \pi_{x,y}(S) &= S(x, y) \\
 \pi_{x,y}(R^-) &= \pi_{y,x}(R) \\
 \pi_{x,y}(R_1 \circ \dots \circ R_n) &= (\exists x_1) \dots (\exists x_{n-1}) \\
 &\quad \left(\pi_{x,x_1}(R_1) \wedge \bigwedge_{i=1}^{n-2} \pi_{x_i,x_{i+1}}(R_{i+1}) \wedge \pi_{x_{n-1},y}(R_n) \right) \\
 \pi(\text{Ref}(R)) &= (\forall x) \pi_{x,x}(R) \\
 \pi(\text{Asy}(R)) &= (\forall x)(\forall y)(\pi_{x,y}(R) \rightarrow \neg \pi_{y,x}(R)) \\
 \pi(\text{Dis}(R_1, R_2)) &= \neg (\exists x)(\exists y)(\pi_{x,y}(R_1) \wedge \pi_{x,y}(R_2))
 \end{aligned}$$

- ...which (interpreted under FOL semantics) coincides with the definition just given.

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- OWL Semantics
- **OWL Profiles**
- Proof Theory
- Tools
- Outlook
- References

- OWL Full – using the RDFS-based semantics
- OWL DL – using the FOL semantics

The OWL 2 documents describe further profiles, which are of polynomial complexity:

- OWL EL (EL++)
- OWL QL (DL Lite_R)
- OWL RL (DLP)

■ allowed:

- subclass axioms with intersection, existential quantification, top, bottom
 - closed classes must have only one member
- property chain axioms, range restrictions (under certain conditions)

■ disallowed:

- negation, disjunction, arbitrary universal quantification, role inverses

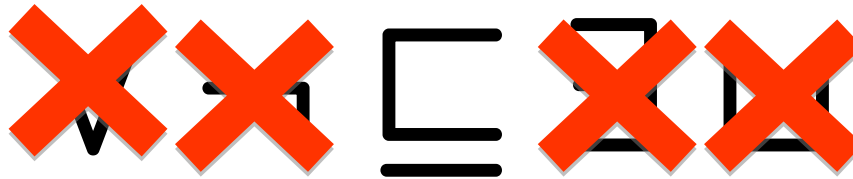
$$\sqcap \exists T \perp \sqsubseteq \sqcap \exists T \perp$$

- Examples: $\exists \text{has.Sorrow} \sqsubseteq \exists \text{has.Liqueur}$; $\top \sqsubseteq \exists \text{hasParent.Person}$
 $\exists \text{married}.\top \sqcap \text{CatholicPriest} \sqsubseteq \perp$; $\text{German} \sqsubseteq \exists \text{knows}\{angela\}$;
 $\text{hasParent} \pm \text{hasParent} \sqsubseteq \text{hasGrandparent}$

- motivated by the question: what fraction of OWL 2 DL can be expressed by rules (with equality)?
- examples:
 - $\exists \text{parentOf}.\exists \text{parentOf}.\top \sqsubseteq \text{Grandfather}$
rule version: $\text{parentOf}(x,y) \wedge \text{parentOf}(y,z) \rightarrow \text{Grandfather}(x)$
 - $\text{Orphan} \sqsubseteq \forall \text{hasParent}.\text{Dead}$
rule version: $\text{Orphan}(x) \wedge \text{hasParent}(x,y) \rightarrow \text{Dead}(y)$
 - $\text{Monogamous} \sqsubseteq \leq 1 \text{married}.\text{Alive}$
rule version:
 $\text{Monogamous}(x) \wedge \text{married}(x,y) \wedge \text{Alive}(y) \wedge \text{married}(x,z) \wedge \text{Alive}(z) \rightarrow y=z$
 - $\text{childOf} \pm \text{childOf} \sqsubseteq \text{grandchildOf}$
rule version: $\text{childOf}(x,y) \wedge \text{childOf}(y,z) \rightarrow \text{grandchildOf}(x,z)$
 - $\text{Disj}(\text{childOf}, \text{parentOf})$
rule version: $\text{childOf}(x,y) \wedge \text{parentOf}(x,y) \rightarrow$

■ syntactic characterization:

- essentially, all axiom types are allowed
- disallow certain constructors on lhs and rhs of subclass statements



- cardinality restrictions: only on rhs and only ≤ 1 and ≤ 0 allowed
 - closed classes: only with one member
- ## ■ Reasoner conformance requires only soundness.

- motivated by the question: what fraction of OWL 2 DL can be captured by standard database technology?
- formally: query answering LOGSPACE w.r.t. data (via translation into SQL)
- allowed:
 - subproperties, domain, range
 - subclass statements with
 - left hand side: class name or expression of type $\exists r.T$
 - right hand side: intersection of class names, expressions of type $\exists r.C$ and negations of lhs expressions
 - no closed classes!
- Example:
 $\exists \text{married}.T \sqsubseteq \neg \text{Free} \sqcap \exists \text{has.Sorrow}$

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- OWL Semantics
- OWL Profiles
- **Proof Theory**
- Tools
- Outlook
- References

- Traditionally using tableaux algorithms (see below)

Alternatives:

- Transformation to disjunctive datalog using basic superposition done for SHIQ
- Naive mapping to Datalog for OWL RL
- Mapping to SQL for OWL QL
- Special-purpose algorithms for OWL EL
e.g. transformation to Datalog

Proof theory Via Tableaux

- Adaptation of FOL tableaux algorithms.
- Problem: OWL is decidable, but FOL tableaux algorithms do not guarantee termination.
- Solution: *blocking*.

DL Tableaux Termination Problem

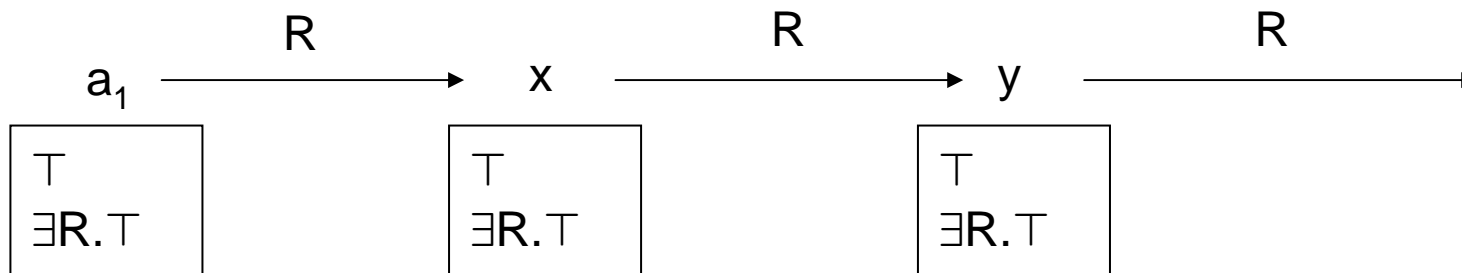
TBox: $\exists R.T$

ABox: $\top(a_1)$

■ **Is satisfiable:**

**Model M contains elements a_1^M, a_2^M, \dots
and $R^M(a_i^M, a_{i+1}^M)$ for all $i \geq 1$.**

■ **But naive tableau does not terminate!**

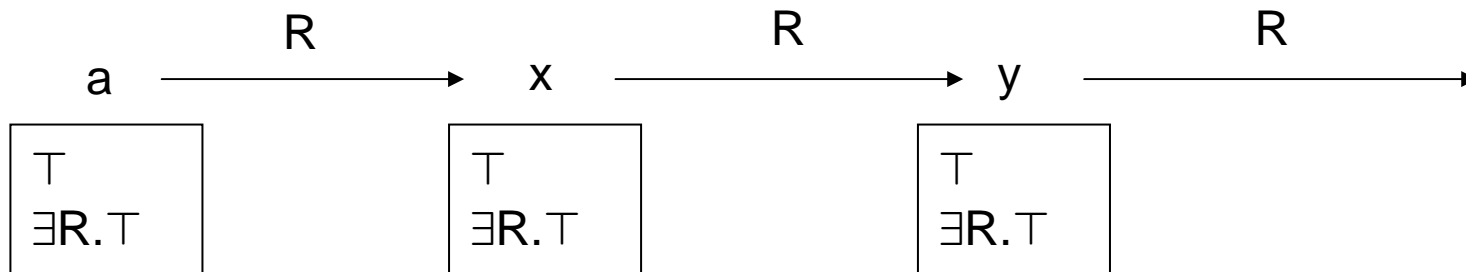


DL Tableaux Termination Problem

Nothing essentially new happens.

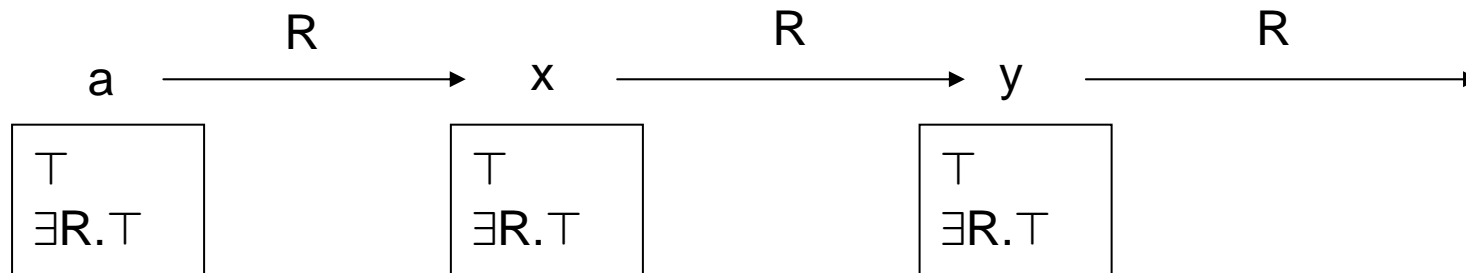
Idea: y does not need to be expanded, because it is basically a copy of x .

⇒ **Blocking**



Blocking (in ALC)

- y is *blocked* (by x) if
 - y is not an individual (but a variable),
 - y is a successor of x and $L(y) \subseteq L(x)$,
 - or an ancestor of y is blocked.



y blocked by x in this example.

Blocking conditions for more expressive DLs are more involved;
the idea is the same.

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- OWL Semantics
- OWL Profiles
- Proof Theory
- **Tools**
- Outlook
- References

OWL tools (incomplete listing)

Reasoner:

■ OWL 2 DL:

- Pellet <http://clarkparsia.com/pellet/>
- HermiT <http://www.hermit-reasoner.com/>

■ OWL 2 EL:

- CEL <http://code.google.com/p/cel/>

■ OWL 2 RL:

- essentially any rule engine

■ OWL 2 QL:

- essentially any SQL engine (with a bit of query rewriting on top)

Editors:

- Protégé
- NeOn Toolkit
- TopBraid Composer

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- OWL Semantics
- OWL Profiles
- Proof Theory
- Tools
- **Outlook**
- References

Some Current Research Questions

- Integrating OWL and Rules
- Inconsistency handling / paraconsistent reasoning
- Local closed world reasoning
- Uncertainty handling (fuzzy / probabilistic)
- Modularization
- Distributedness
- Belief Revision (Ontology Evolution)
- Abduction/Explanation/Justification
- Approximate Reasoning
- Ontology Learning
- Modelling / Design Patterns
- Ontology Engineering (Modelling Processes)
- Interfaces (GUIs, Constrained Natural Language, etc.)

- **Several major conferences on Semantic Web:**
 - ISWC (>600), ESWC (>300), WWW Semantic Web track, IJCAI Semantic Web track, etc.
- **Semantic Web languages taught in many university courses world-wide.**
 - Becomes established topic.
- **Industrial uptake currently happening**
 - e.g. OWL reasoners by IBM, ORACLE
 - many application studies by major IT companies
 - considerable number of spin-offs
 - venture capital (VULCAN Inc.)
- **Considerable uptake in the life sciences**

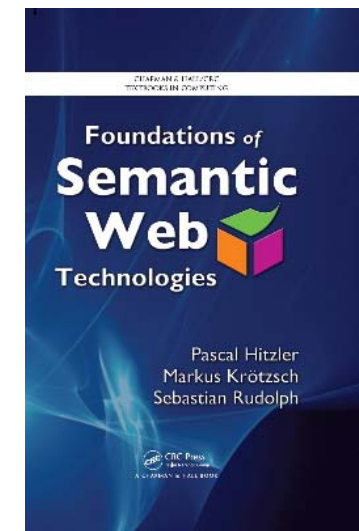
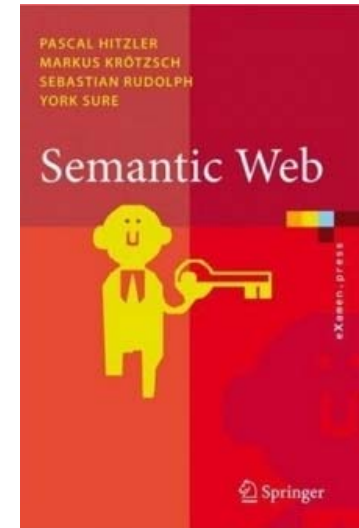
- **Substantial project funding (EU, NIH, etc.)**

- OWL – Basic Ideas
- OWL As the Description Logic SROIQ(D)
- Different Perspectives on OWL
- Expressivity Examples: Rules in OWL
- OWL Semantics
- OWL Profiles
- Proof Theory
- Tools
- Outlook
- **References**

- **W3C OWL Working Group, OWL 2 Web Ontology Language: Document Overview. <http://www.w3.org/TR/owl2-overview/>**
- **Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter Patel-Schneider, Sebastian Rudolph, OWL 2 Web Ontology Language: Primer. <http://www.w3.org/TR/owl2-primer/>**
- **Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider, The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2nd edition, 2007.**

Main References – Textbooks

- **Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure, Semantic Web – Grundlagen.**
Springer, 2008.
<http://www.semantic-web-grundlagen.de/>
(In German.)
(Does not cover OWL 2.)
- **Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, Foundations of Semantic Web Technologies.**
Chapman & Hall/CRC, 2009.
<http://www.semantic-web-book.org/wiki/FOST>
(Ask for a flyer from us.)



Further References

- DL complexity calculator: <http://www.cs.man.ac.uk/~ezolin/dl/>
- Markus Krötzsch, Sebastian Rudolph, Pascal Hitzler, Description Logic Rules. In Malik Ghallab, Constantine D. Spyropoulos, Nikos Fakotakis, Nikos Avouris, eds.: Proceedings of the 18th European Conference on Artificial Intelligence (ECAI-08), pp. 80–84. IOS Press 2008.

Thanks!

http://semantic-web-grundlagen.de/wiki/IJCAI-09_Tutorial